

(12) **UK Patent Application** (19) **GB** (11) **2 350 533** (13) **A**

(43) Date of A Publication 29.11.2000

(21) Application No 9912575.9

(22) Date of Filing 28.05.1999

(71) Applicant(s)  
**Mitel Corporation**  
 (Incorporated in Canada - Ontario)  
 PO Box 13089, 350 Legget Drive, Kanata, Ontario,  
 K2K 1X3, Canada

(72) Inventor(s)  
**Robert Geoffrey Wood**

(74) Agent and/or Address for Service  
**Cruikshank & Fairweather**  
 19 Royal Exchange Square, GLASGOW, G1 3AE,  
 United Kingdom

(51) INT CL<sup>7</sup>  
**H04L 25/02, G06F 5/06**

(52) UK CL (Edition R )  
**H4P PF**  
**U1S S2124**

(56) Documents Cited  
**GB 2331678 A** **WO 97/17777 A1** **JP 060268692 A**  
**US 5765187 A**

(58) Field of Search  
 UK CL (Edition Q ) **G4A AKB1, H4P PF PT**  
 INT CL<sup>6</sup> **G06F 5/06, H04L 25/02 25/36 25/50**  
 Online: **WPI, EPODOC, JAPIO**

(54) Abstract Title  
**Avoiding underflow or overflow in a circular buffer**

(57) A method is provided for avoiding data loss in a data packet switch which utilizes a circular data buffer. If the data is received at a faster rate than it is read out of the buffer the data read-out pointer is adjusted by incrementing it to skip, or drop, the next sample. If the data is received at a slower rate than it is read out of the buffer, then the read-out pointer is adjusted by decrementing it to repeat the previous sample.

GB 2 350 533 A

At least one drawing originally filed was informal and the print reproduced here is taken from a later filed formal copy.

This print takes account of replacement documents submitted after the date of filing to enable the application to comply with the formal requirements of the Patents Rules 1995

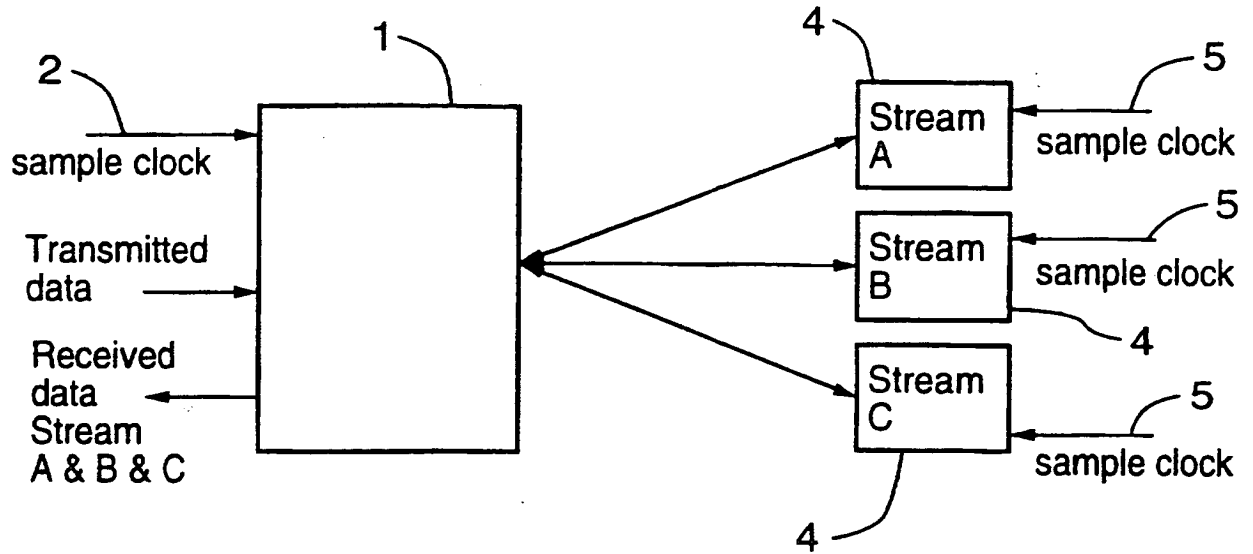


FIG.1

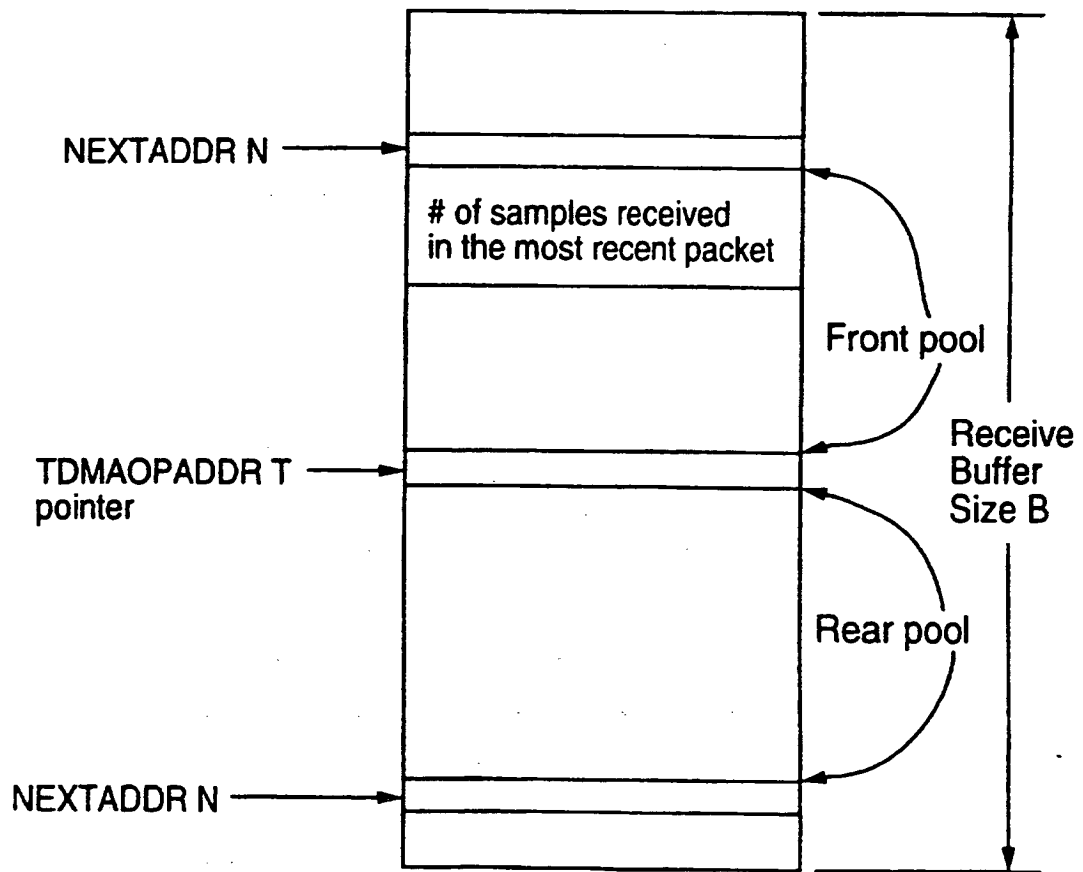


FIG.2

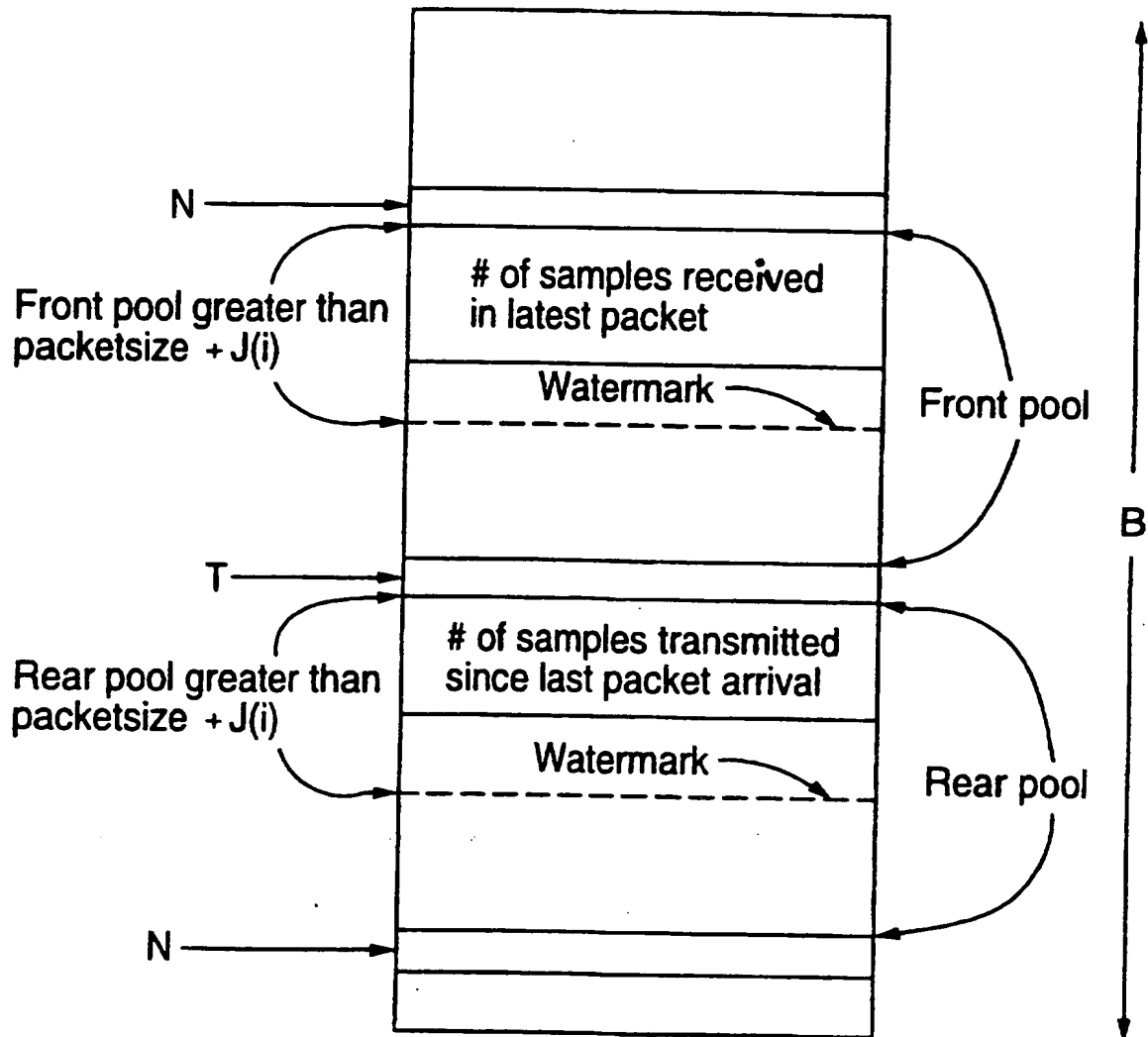


FIG.3

**METHOD TO CONTROL DATA RECEPTION BUFFERS FOR PACKETIZED  
VOICE CHANNELS**

**FIELD OF THE INVENTION**

5

This invention relates in general to data transfer systems and more specifically to a method for controlling data transfer into and out of a circular buffer so as to prevent buffer overflow and underflow.

**10 BACKGROUND OF THE INVENTION**

With the advent of Voice-Over-IP (VoIP) communication systems, specialized hardware has been developed to convert synchronous TDM streams of voice data to voice packets for transmission via IP (Internet Protocol) over a LAN (Local Area  
15 Network), and vice versa. In order to accommodate variations in transmission rates into and out of such E2T devices (Ethernet-to-TDM), software algorithms have been developed to prevent data loss where the receive and transmit data streams are not synchronized. Software solutions such as are known in the prior art contribute to system complexity and loss of speed for real time applications (such as the  
20 transmission of voice traffic).

**SUMMARY OF THE INVENTION**

According to the present invention, a method is provided for avoiding data  
25 loss in a data packet switch which utilizes a circular data buffer. If the data is received at a faster rate than it is read out of the buffer, then the transmitter is running at a higher frequency than the receiver and the buffer will soon overflow. Therefore, according to one aspect of the invention the data read-out pointer is adjusted by incrementing it to skip, or drop, the next sample. If the data is received at a slower  
30 rate than it is read out of the buffer, then the transmitter is running at a lower frequency than the receiver and the buffer will soon underflow. Therefore, according to another aspect of the invention, the read-out pointer is adjusted by decrementing it to repeat the previous sample. The method of controlling the buffer read-out pointer

according to the present invention, is implemented in hardware thereby reducing system complexity and improving speed relative to prior art software solutions.

5 The method according to the present invention accommodates dynamically varying packet sizes and permits a reduction in the receive buffer size, thereby resulting in savings in buffer memory and delay and latency over prior art software implementations. Also, the method of the present invention is dynamically adjustable according the network jitter, as defined by the Real Time Protocol operations specified by the IETF's RFC1889. The algorithm of the present invention does not  
10 add significant distortion or inject appreciable noise into the voice data stream, and allows for packets containing silence to be suppressed (i.e. the packets are not transmitted, thereby allowing the TDM data to be continually read out of the buffer at the local sample rate).

## 15 BRIEF DESCRIPTION OF THE DRAWINGS

Various embodiments of the present invention are described below with reference to the drawings in which:

20 Figure 1 is a block diagram showing a data packet transmission system according to the prior art;

Figure 2 is a schematic representation of a circular buffer for use in the data packet transmission system of Figure 1; and

25

Figure 3 is a further schematic representation of the circular buffer of Figure 2 with notations for illustrating the buffer control method according to the present invention.

## 30 DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

Figure 1 shows a device (1) that forms TDM data streams, normally voice streams, into packets and transmits them over a packet network. In operation, the

device (1) receives transmitted data at a local sample rate, defined by a local frame pulse (2), places the TDM data into appropriate buffers (e.g. one buffer per voice stream), and transmits entire buffer contents as a data packet when a predetermined number of samples has been collected in the buffer(s). All transmitted voice streams  
 5 are transmitted at the same, locally defined, rate.

The device (1) also receives packets of TDM data streams from a multiplicity of external devices (4). These packets have data rates referenced to synchronous sources, defined by local sample clocks (5) which are independent and unrelated to  
 10 each other and the local frame pulse (2), but have the same nominal frequency and data rate as their respective local references (5).

When a packet from a particular one of the sources (4) is received, it is stored temporarily in a dedicated local memory buffer within the device (1). Single samples  
 15 are extracted from this buffer at the local data rate (defined by the frame pulse (2)), for transmission over the local TDM channel (i.e. the received data stream A, B & C in Figure 1). The data streams from sources A, B and C have different data rates but are output on the received TDM data stream output of the device (1) at the same data rate, which is governed by the local frame pulse (2).

20

Differences in the data rates into and out of each receive buffer within device (1) can result in the buffers running out of data or overfilling with data.

Figure 2 is a representation of a circular buffer and illustrates the various  
 25 attributes of that buffer. The circular buffer of Figure 2 is of a fixed size (B samples). Data packets are written into the buffer when received, starting at the location pointed to by the pointer NEXTADDR. Data is read out, one sample at a time, from the location pointed to by a TDMAOPADDR pointer, which is incremented after each sample is read so as to point to the next sample. Since the buffer is circular, the  
 30 pointer NEXTADDR could have a value larger or smaller than TDMAOPADDR, as shown.

The number of samples in the buffer to be transmitted is defined as the "front pool". If data is received too slowly, then the front pool gradually shrinks until TDMAOPADDR equals NEXTADDR at which time there is no more data to transmit.

5

The amount of space available in the buffer to receive new packets is defined as the "rear pool". If this space becomes less than the size of a packet, then the new packet will overwrite data that has yet to be transmitted.

10 The method according to the present invention prevents either of these two conditions from occurring, by adjusting the pointer TDMAOPADDR to ensure adequate front and rear pools.

With reference to Figure 3, data is read out of the buffer at the local receiver  
15 TDM sample rate, one sample at a time, with the result that the TDMAOPADDR pointer (T), is incremented so as to point to the next sample to be read. If the data is received at the same rate as it is read out of the buffer, then transmitter and receiver are operating at the same frequency. If the data is received at a faster rate than it is read out of the buffer, then the transmitter is running at a higher frequency than the  
20 receiver and the buffer will soon overflow, (i.e. when the rear pool becomes zero). To avoid this happening, according to the present invention the TDMAOPADDR pointer is adjusted by incrementing it to skip, or drop, the next sample. On the other hand, if the data is received at a slower rate than it is read out of the buffer, then the transmitter is running at a lower frequency than the receiver and the buffer will soon  
25 underflow, (i.e. when the front pool becomes zero). To avoid this happening, according to the present invention the TDMAOPADDR pointer is adjusted by decrementing it to repeat the previous sample. These additional adjustments to TDMAOPADDR are made each time a packet is received, thereby allowing for an adequate adjustment rate and the implementation of simple silence suppression  
30 schemes. In the latter case, if no packets are received due to silence, then TDMAOPADDR will continue being incremented at the local sample rate, (i.e. effectively free-running).

A front pool watermark is defined as  $N - \text{packet size} - J(i)$  where  $N$  is the starting address for writing the next packet; "packet size" is the number of samples contained in the packet; and  $J(i)$  is the Real Time Protocol packet inter-arrival jitter as calculated according to the Internet Engineering Task Force's RFC1889, section 6.3.1.

5

A rear pool watermark is defined as  $T - \text{packet size} - J(i)$  where  $T$  is the starting address of the next sample to be read out of the buffer, at the local receiver's sample rate; "packet size" is the number of samples contained in the packet; and  $J(i)$  is the Real Time Protocol packet inter-arrival jitter as calculated according to the Internet Engineering Task Force's RFC1889, section 6.3.1.

10

For an arbitrary buffer size, the actual size of the front pool is calculated by the algorithm  $FP = N - T$ . If  $FP$  is negative, then  $FP = FP + B$ . For a buffer size which is a binary multiple and with  $N$  and  $T$  being integers of width equivalent to the minimum number of bits required to fully address the entire buffer, then, ignoring overflow, the equation reduces to  $FP = N - T$ . Similarly, for the rear pool,  $RP = T - N$ .

15

The following is a section of verilog code for implementing the algorithm according to the present invention, for a buffer size of 1k samples:

20

25

```

reg [9:0] FP, RP, N, T;

FP = N - T;

RP = T - N;

//Increment TDMAOPADDR to point to next TDM data sample

T = T + 1;

//test if TDMAOPADDR pointer adjustment is required
if ((FP) < ((J >> 4) + packet_size)) T = T - 1;
if ((RP) < ((J >> 4) + packet_size)) T = T + 1;

```

It will be appreciated that, although a particular embodiment of the invention has been described and illustrated in detail, various changes and modifications may be made.

For example, buffers can be of arbitrary sizes. Thus, for a buffer of arbitrary size (but no greater than 1K), the algorithm is described by the following verilog code:

```

reg [10:0] FP, RP;
reg [9:0] N, T, B;
FP = {1b'0,N} - {1b'0,T};
if (FP[10]) FP = FP + B;
RP = {1b'0,T} - {1b'0,N};
if (RP[10]) RP = RP + B;
//Increment TDMAOPADDR to point to next TDM data sample
T = T + 1;
//test if TDMAOPADDR pointer adjustment is required
if ((FP[9:0]) < ((J >> 4) + packet_size)) T = T - 1;
if ((RP[9:0]) < ((J >> 4) + packet_size)) T = T + 1;
```

Also, packet size can be either a fixed quantity, with all packets containing the same number of samples, or the packet size can be dynamic, (i.e. varying from packet to packet). However, the value, packet\_size, is provided by the packet reception circuitry in a well known manner.

When the method of the present invention is used to control buffers for packets which are not RTP encapsulated, then the jitter variable, J, is not available. The variable storage element for J in the verilog code implementations above, can therefore be set to a predetermined value which remains constant for all packets. The value set will generally be larger than the maximum network jitter expected and can be larger or smaller than single, or multiple, packets.

All such changes and modifications may be made without departing from the sphere and scope of the invention as defined by the claims appended hereto.

What is claimed is:

1. A method for controlling data transfer into and out of a circular buffer, comprising the steps of:

5

receiving and writing data packets into said buffer starting at a first address and at incrementally increasing addresses thereafter;

10

reading and transmitting data packets out of said buffer starting at a second address and at incrementally increasing addresses thereafter;

15

in the event that said first address is greater than said second address by less than a predetermined amount then decrementing said second address before reading and transmitting of said data packets; and

in the event that said second address is greater than said first address by less than a predetermined amount then incrementing said second address before reading and transmitting of said data packets.

20

2. The method of claim 1, wherein each of said data packets is characterized by a packet size and an inter-arrival jitter value, and wherein said predetermined amount is equivalent to said packet size plus said inter-arrival jitter value.

25

3. The method of claim 1, wherein each of said data packets is characterized by a packet size and wherein said predetermined amount is equivalent to said packet size plus a constant value which is larger than maximum expected network jitter associated with said data packets.



Application No: GB 9912575.9  
Claims searched: 1-3

Examiner: Steven Davies  
Date of search: 9 December 1999

**Patents Act 1977**  
**Search Report under Section 17**

**Databases searched:**

UK Patent Office collections, including GB, EP, WO & US patent specifications, in:

UK Cl (Ed.Q): G4A AKB1 ; H4P PF, PT

Int Cl (Ed.6): G06F 5/06 ; H04L 25/02, 25/36, 25/50

Other: Online databases: WPI, EPODOC, JAPIO

**Documents considered to be relevant:**

Category	Identity of document and relevant passage	Relevant to claims
A	GB 2331678 A (DAEWOO)	
A	WO 97/17777 A1 (ERICSSON)	
A	US 5765187 (SHIMIZU et al)	
A	JP 060268692 A (NEC) see JAPIO abstract	

X Document indicating lack of novelty or inventive step  
Y Document indicating lack of inventive step if combined with one or more other documents of same category.  
& Member of the same patent family

A Document indicating technological background and/or state of the art.  
P Document published on or after the declared priority date but before the filing date of this invention.  
E Patent document published on or after, but with priority date earlier than, the filing date of this application.

**This Page is Inserted by IFW Indexing and Scanning  
Operations and is not part of the Official Record**

**BEST AVAILABLE IMAGES**

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☒ **BLACK BORDERS**
- ☐ **IMAGE CUT OFF AT TOP, BOTTOM OR SIDES**
- ☐ **FADED TEXT OR DRAWING**
- ☐ **BLURRED OR ILLEGIBLE TEXT OR DRAWING**
- ☐ **SKEWED/SLANTED IMAGES**
- ☐ **COLOR OR BLACK AND WHITE PHOTOGRAPHS**
- ☐ **GRAY SCALE DOCUMENTS**
- ☐ **LINES OR MARKS ON ORIGINAL DOCUMENT**
- ☐ **REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY**
- ☐ **OTHER:** \_\_\_\_\_

**IMAGES ARE BEST AVAILABLE COPY.**

**As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.**

**THIS PAGE BLANK (USPTO)**